BUNDESREPUBLIK DEUTSCHLAND

Prioritätsbescheinigung über die Einreichung einer Patentanmeldung

Aktenzeichen: 101 47 411.3

Anmeldetag: 26. September 2001

Anmelder/Inhaber: Siemens Aktiengesellschaft, 80333 München/DE

Bezeichnung: System und Verfahren zur Zugriffskoordinierung in Kommunikationssystemen, insbesondere Real-time Ethernet

IPC: H 04 L 12/407

Die angehefteten Stücke sind eine richtige und genaue Wiedergabe der ursprünglichen Unterlagen dieser Patentanmeldung.

München, den 23. Januar 2004
Deutsches Patent- und Markenamt
Der Präsident
Im Auftrag

Hintermeier

A 9161
06/00
EDV-L

Beschreibung

System und Verfahren zur Zugriffskoordinierung in Kommunikationssystemen, insbesondere Real-time Ethernet

Die Erfindung bezieht sich auf ein System und Verfahren zur Zugriffskoordinierung in Kommunikationssystemen, insbesondere Real-time Ethernet.

Unter einem synchronen, getakteten Kommunikationssystem mit Äquidistanz-Eigenschaften versteht man ein System aus wenigstens zwei Teilnehmern, die über ein Datennetz zum Zweck des gegenseitigen Austausches von Daten bzw. der gegenseitigen Übertragung von Daten miteinander verbunden sind. Dabei erfolgt der Datenaustausch zyklisch in äquidistanten Kommunikationszyklen, die durch den vom System verwendeten Kommunikationstakt vorgegeben werden. Teilnehmer sind beispielsweise zentrale Automatisierungsgeräte, Programmier-, Projektierungs- oder Bediengeräte, Peripheriegeräte wie z.B. Ein-/Ausgabe-Baugruppen, Antriebe, Aktoren, Sensoren, speicherprogrammierbare Steuerungen (SPS) oder andere Kontrolleinheiten, Computer, oder Maschinen, die elektronische Daten mit anderen Maschinen austauschen, insbesondere Daten von anderen Maschinen verarbeiten. Teilnehmer werden auch Netzwerkknoten oder Knoten genannt. Unter Kontrolleinheiten werden im folgenden Regler- oder Steuerungseinheiten jeglicher Art verstanden, aber auch beispielsweise Switches und/oder Switch-Controller. Als Datennetze werden beispielsweise Bussysteme wie z.B. Feldbus, Profibus, Ethernet, Industrial Ethernet, FireWire oder auch PC-interne Bussysteme (PCI), etc., insbesondere aber auch isochrones Realtime Ethernet verwendet.

Datennetze ermöglichen die Kommunikation zwischen mehreren Teilnehmern durch die Vernetzung, also Verbindung der einzelnen Teilnehmer untereinander. Kommunikation bedeutet dabei die Übertragung von Daten zwischen den Teilnehmern. Die zu übertragenden Daten werden dabei als Datentelegramme verschickt,

d.h. die Daten werden zu mehreren Paketen zusammengepackt und in dieser Form über das Datennetz an den entsprechenden Empfänger gesendet. Man spricht deshalb auch von Datenpaketen. Der Begriff Übertragung von Daten wird dabei in diesem Dokument völlig synonym zur oben erwähnten Übertragung von Datentelegrammen oder Datenpaketen verwendet.

In verteilten Automatisierungssystemen, beispielsweise im Bereich Antriebstechnik, müssen bestimmte Daten zu bestimmten Zeiten bei den dafür bestimmten Teilnehmern eintreffen und von den Empfängern verarbeitet werden. Man spricht dabei von echtzeitkritischen Daten bzw. Datenverkehr, da ein nicht rechtzeitiges Eintreffen der Daten am Bestimmungsort zu unerwünschten Resultaten beim Teilnehmer führt, im Gegensatz zur nicht echtzeitkritischen, beispielsweise inter- bzw. intranetbasierten Datenkommunikation. Gemäss IEC 61491, EN61491 SERCOS interface - Technische Kurzbeschreibung (http://www.sercos.de/deutsch/index_deutsch.htm) kann ein erfolgreicher echtzeitkritischer Datenverkehr der genannten Art in verteilten Automatisierungssystemen gewährleistet werden.

Automatisierungskomponenten (z.B. Steuerungen, Antriebe,...) verfügen heute im Allgemeinen über eine Schnittstelle zu einem zyklisch getakteten Kommunikationssystem. Eine Ablaufebene der Automatisierungskomponente (Fast-cycle) (z.B. Lageregelung in einer Steuerung, Drehmomentregelung eines Antriebs) ist auf den Kommunikationszyklus synchronisiert. Dadurch wird der Kommunikationstakt festgelegt. Andere, niederperformante Algorithmen (Slow-cycle) (z.B. Temperaturregelungen) der Automatisierungskomponente können ebenfalls nur über diesen Kommunikationstakt mit anderen Komponenten (z.B. Binärschalter für Lüfter, Pumpen,...) kommunizieren, obwohl ein langsamerer Zyklus ausreichend wäre. Durch Verwendung nur eines Kommunikationstaktes zur Übertragung von allen Informationen im System entstehen hohe Anforderungen an die Bandbreite der Übertragungsstrecke.

Beim Betrieb einer Befehlseinheit (Command-Interface) an einem Multi-Master-System, können mehrere Teilnehmer (Master) gleichzeitig oder nacheinander, in jedem Fall jedoch unkoordiniert auf eine Befehlseinheit zugreifen. Eine Koordinierung der einzelnen Teilnehmer (mehrere Prozessoren oder unterschiedliche Anwendungen eines Prozessors) auf das Command-Interface ist hierbei nötig um die Übergabe und Bearbeitung der Befehle am Command-Interface zu gewährleisten.

Die Koordination wurde bisher auf Software-Ebene (Treiber) realisiert, indem für die Dauer der Bearbeitung die Anwendungen durch Interrupt-Sperren blockiert wurden. Auf Hardware-Ebene konnte durch ein Bus-Locking ein weiterer Zugriff auf das Command-Interface verhindert werden. Somit wurde ebenfalls die Software für die Dauer der Bearbeitung angehalten. Nachteil:

- Auch wenn eine Anwendung keinen Zugriff auf das Command-Interface durchführen will, wird aufgrund der Bus-Locking-Mechanismen die Softwarebearbeitung unterbrochen.
- Durch die Interrupt-Sperre kann für die Dauer der Befehlsbearbeitung ein auftretendes Interrupt-Ereignis nicht sofort bearbeitet werden, die Interrupt-Routine wird verspätet ausgeführt.

Über das von mehreren Teilnehmern gemeinsam genutzte Interface, kann die Befehlseinheit angewiesen werden bestimmte Grundoperationen auszuführen. Dabei werden alle Teilnehmer als gleichwertige Mitglieder des Command-Interfaces betrachtet. Die Befehlsübergabe ans Command-Interface erfolgt dabei nicht direkt durch Übergabe der Befehlsparameter, sondern indirekt, indem lediglich die Adresse auf eine Befehlsstruktur übergeben wird. Die Übergabe dieser Adresse ist ein atomarer Schreibzugriff auf das Command-Interface, was keine Bus-Lock-Mechanismen oder Interrupt-Sperren benötigt. Die prinzipiellen Unterschiede einer indirekten Befehlsübergabe am Command-Interface gegenüber dem direkten Übergeben von Parametern sind:

- Kommandos können zu ihrer Ausführung mehr Informationen als nur ein 32-Bit Wort benötigen.
- Sie haben eine begrenzte Ausführungsdauer. Ihre Ausführung kann unterschiedlich viel Information zurückliefern.
- Kommandos können von mehreren Instanzen abgegeben werden, die nicht notwendigerweise untereinander koordiniert sein müssen. Die Instanzen können über mehrere Tasks sowohl eines oder mehrerer Teilnehmer verteilt sein.
- Mehrere, auch gleiche Kommandos können direkt nacheinander am Command-Interface abgesetzt werden, ohne die Ausführung jedes einzelnen Kommandos abwarten zu müssen.

Um die beschriebene allgemeine Problematik eines Multiprozessor- und Multitaskingsystems zu umgehen, sind für das Command-Interface bestimmte Voraussetzungen hinsichtlich Adressierung und Übergabe der Befehle notwendig. Diese sind:

- ausschließlich indirekte Adressierung der Befehle
- Bestätigung übernommener Befehle durch das Command-Interface

Damit wird erreicht, dass:

- keinerlei Anforderungen an das Bussystem bezüglich Locken bzw. Blockierung gestellt werden muss bzw. der Bus nie über eine einzelne Lese- oder Schreiboperation hinaus gelockt werden muss,
- beliebig viel Information hin und zurück übertragen werden kann,
- keine Koordination wie z.B. Interruptsperren zwischen den aufrufenden Instanzen notwendig ist.

Zur Vermeidung von Bus-Lock-Zeiten müssen alle von den Teilnehmern abgesetzten Befehle in einem Schreibzyklus dem Command-Interface übergeben werden. Befehle mit mehreren Operanden hingegen würden mehrfache Schreibzugriffe auf das Command-Interface bedeuten. Um dies zu verhindern, wird dem Command-Interface lediglich ein Zeiger (Adresse) auf einen Speicherbereich übergeben. Von dort werden dann die Befehlsdaten

aus einer definierten Befehlsstruktur vom Command-Interface ausgelesen, interpretiert und der entsprechenden Ausführungseinheit zur Bearbeitung übergeben.

Die aktiven Busteilnehmer können unabhängig voneinander Schreibzugriffe auf das Command-Interface durchführen. Auch die Multitasking-Fähigkeiten eines Teilnehmers erlaubt nicht geordnete (gemischte) Zugriffe der verschiedenen Anwendungen auf das Command-Interface. Um die einzelnen Tasks eines Teilnehmers nicht durch Interrupt-Sperren gegenseitig zu verriegeln, sind im Command-Interface Mechanismen implementiert, die einen wahlfreien Zugriff aller Anwendungen und deren Teilnehmer in verschiedenster Reihenfolge und zu unterschiedlichen Zeiten auf das Interface zulassen. Generell gilt, die Anwendungen müssen erkennen können, ob ihre Zeiger auf eine Befehlsstruktur, die in das Command-Interface geschrieben wurde, übernommen worden sind. Sind gelesene und geschriebene Adresse identisch, so wurde das Kommando am Command-Interface akzeptiert. Sind sie ungleich, so können hierfür zwei Möglichkeiten verantwortlich sein:

- Das Kommando wurde nicht akzeptiert, weil die Übernahme eines früheren Kommandos noch nicht abgeschlossen war.
- Das Kommando wurde bereits übernommen, aber zwischen dem Schreiben und Lesen wurde bereits ein weiteres Kommando übernommen. Diese Situation kann sich sowohl mit einem Teilnehmer (Kernel mit preemptiven Multitasking, wenn dies nicht vom Betriebssystem verwaltet wird) als auch bei Systemen mit mehreren physikalischen Teilnehmern ergeben.

Daher werden von der Hardware zwei Mechanismen unterstützt, die beide von der Software zur eindeutigen Erkennung genutzt werden müssen:

- Rücklesen der gerade geschriebenen Daten (Adresse auf Befehlsstruktur) aus dem Command-Interface. Dabei wird erkannt, ob die Daten des Teilnehmers am Command-Interface übernommen worden sind. Ist dies der Fall, so wird der Befehl über das Command-Interface ausgeführt, wenn nicht dann wurden bereits vorab die Daten eines zweiten Teilneh-

mers ins Interface eingetragen, so dass ein Command des ersten Teilnehmers nicht angenommen werden konnte.

- Bestätigung der übernommenen Daten. Werden zwischen Beschreiben und Rücklesen der Adresse folgende Aktionen am Command-Interface ausgeführt:
  - Übernahme des Befehls durch das Command-Interface
  - Bestätigen der Übernahme durch ein „Acknowledge"-Feld in der Befehlsstruktur durch das Command-Interface
  - Beschreiben des Command-Interfaces mit neuen Adressdaten des zweiten Teilnehmers

  so stimmen geschriebene und rückgelesene Daten nicht mehr überein. Die Übernahme eines Befehls kann jetzt lediglich am „Acknowledge"-Feld der Befehlsstruktur eindeutig erkannt werden.

Um ein critical race auszuschließen, wird immer erst das „Acknowledge"-Feld gesetzt, bevor das Command-Interface wieder beschreibbar wird. Die Übernahme eines Kommandos ist in der Regel mit der Einkettung der Befehlsstruktur in eine Command-Liste abgeschlossen, sie erfolgt also schnell. Eine Optimierung auf eine möglichst kurze und garantierte Übernahmezeit hat Priorität, um bei einem erneuten Schreibzugriff auf das Command-Interface keine unnötigen Wartezeiten in der Abarbeitung der Software zu erhalten.

Aufbau des Command-Interface

Über ein gemeinsam genutztes Command-Register, im folgenden auch Eingangs-Register genannt, wird dem Command-Interface der Befehl bzw. die Befehlsfolgen beider Teilnehmer indirekt übergeben. Dabei wird im Command-Register die Adresse (Zeiger) auf eine Befehlsstruktur abgelegt. Die Struktur selbst enthält alle für die Abarbeitung des Befehls notwendigen Daten. Um bei langer Befehlsbearbeitung das Command-Register nicht für die gesamte Befehlsbearbeitungs-Dauer zu blockieren, wird die übergebene Befehlsstruktur in einer Command-Liste eingekettet. Die Befehlsstruktur enthält dafür einen „next"-Zeiger, mit deren Hilfe noch nicht bearbeitete Struk-

turen verkettet werden können. Der „next"-Zeiger enthält die Adresse der nächsten noch nicht bearbeiteten Befehlsstruktur. Über diese Verkettung wird eine Zwischenspeicherung der Befehlsstrukturen in der Command-Liste erreicht.
Dadurch wird eine Entkopplung zwischen Befehlsbearbeitung und Befehlsübergabe erreicht.
Erfolgt ein Schreibzugriff von einem der Teilnehmer auf das Register, wird die Übernahme weiterer Daten am Command-Register solange verhindert bis das übergebene ICW in der Command-Liste eingekettet ist. Erst nachdem das Command-Interface die Übernahme des Befehls durch Setzen des entsprechenden „Acknowledge"-Feldes im ICW bestätigt, wird ein neuer Schreibzugriff auf das Command-Register akzeptiert.

Nachdem die über das Command-Register übergebenen Befehle abgearbeitet worden sind, werden die zugehörigen Befehlsstrukturen dem Anwender zurückgegeben. Dabei ist für jeden Teilnehmer ein getrenntes Return-Register, im folgenden auch Ausgangs-Register genannt, vorhanden, über das die Befehlsstrukturen übergegeben werden. Um keine schritthaltende Abholung der Befehlsstrukturen am Return-Register durch die Teilnehmer gewährleisten zu müssen, werden die bearbeiteten Strukturen in getrennte teilnehmerabhängige Return-Listen eingekettet. Die Rückgabe jeder bearbeiteten Struktur, wird dem jeweiligen Teilnehmer bekanntgemacht. Die verketteten Befehlsstrukturen werden dabei über das Return-Register zurückgegebenen.

Zusammenfassend gilt:

- Beide Teilnehmer setzen ihre Befehle über ein Command-Register ab.
- Die Befehle werden in einer Command-Listen eingekettet.
- Eine Rückgabe der Befehle erfolgt über getrennte Return-Register und damit auch über getrennte Return-Listen für jeden Teilnehmer.

Wird von der Software ein Befehl abgesetzt, so übergibt der Teilnehmer eine entsprechende Befehlsstruktur durch einen

Schreibzugriff auf das Command-Register an das Command-Interface. Als Folge des Schreibvorgangs auf das Command-Registers werden solange keine Daten nachfolgender Schreibzugriffe akzeptiert, bis die übergebene Struktur vom Command-Interface in die gemeinsame Command-Liste eingekettet und im „Acknowledge"-Feld der Struktur die Übernahme bestätigt wurde. Lesezugriffe auf das Command-Register sind jedoch weiterhin erlaubt. Um zu kontrollieren, ob der gerade eingeschriebene Wert auch ins Command-Register eingetragen worden ist, werden die Daten des Registers wieder zurückgelesen. Stimmen Schreib- und Lesedaten überein, so wurde der Schreibzyklus auf das Command-Register ausgeführt.

Bestehen zwischen den geschriebenen und gelesenen Daten jedoch Unterschiede, so können hierfür zwei Gründe verantwortlich sein:

- Im Zeitraum zwischen dem Schreib- und Lesezyklus eines Teilnehmers 1, wurde die ins Command-Register eingetragene Befehlsstruktur bereits in die Command-Liste übernommen und die Bestätigung im „Acknowledge"-Feld der Struktur gesetzt. Das Einschreiben neuer Daten in das Command-Register wurde daraufhin freigegeben. Noch bevor der Teilnehmer 1 seinen Lesezyklus durchführt, konnte der Teilnehmer 2 einen Schreibzugriff auf das Command-Register absetzen.
- Vor dem Beschreiben des Command-Registers durch einen Teilnehmer 1 wurde bereits vom Teilnehmer 2 ein Schreibzugriff abgesetzt. Der Eintrag der nachfolgenden Adressdaten des Teilnehmers 1 wird dadurch verhindert.

Treten Unterschiede zwischen den Schreib- und Lesedaten auf, so muss der Teilnehmer stets das „Acknowledge"-Feld des Befehlsstruktur analysieren. Ist die Bestätigung in „Acknowledge"-Feld der Struktur gesetzt, so wurde die Befehlsstruktur übernommen und in die Command-Liste eingekettet. Die nicht bestätigte Übernahme der Struktur führt zu einem zyklischen Schreibzugriff auf das Command-Register (Polling).

Der Ablauf der Befehlsübergabe, sowie die Verkopplung zwischen Hard- und Software wird in folgender Abbildung deutlich:
Wahlfreier, unkoordinierter Zugriff mehrerer Teilnehmer auf eine Befehls-Schnittstelle (Command-Interface), ohne:

- die Befehlsbearbeitung abwarten zu müssen
- Sperr-Mechanismen in Soft- oder Hardware, für die Dauer der Befehlsbearbeitung vorsehen zu müssen

Weiterhin können durch das vorgestellte Verfahren:

- eine beliebige Anzahl an Befehlen abgesetzt werden, die Reihenfolge der Abarbeitung wird dabei immer eingehalten
- eine beliebige Menge an Übergabe-Parametern (Operanden) übergeben werden
- eine beliebige Menge an Rückgabe-Parametern (Ergebnissen) zurückgegeben werden

Konzept einer Kommando-Schnittstelle.

Von besonderem Vorteil ist es darüber hinaus, dass die offenbarten Verfahren in Automatisierungssystemen, insbesondere bei und in Verpackungsmaschinen, Pressen, Kunststoffspritzmaschinen, Textilmaschinen, Druckmaschinen, Werkzeugmaschinen, Robotor, Handlingssystemen, Holzverarbeitungsmaschinen, Glasverarbeitungsmaschinen, Keramikverarbeitungsmaschinen sowie Hebezeugen eingesetzt bzw. verwendet werden können.

Im Weiteren werden bevorzugte Ausführungsbeispiele der Erfindung mit Bezugnahme auf die Zeichnungen näher erläutert. Es zeigen:

Fig. 1: Beispiel einer Verkettung von Befehlsstrukturen

Fig. 2: Aufbau des Command-Interface

Fig. 3: Hard- und Softwareablauf am Command-Interface

Patentansprüche

1. Verfahren zur Zugriffskoordinierung in Kommunikationssystemen mit wenigstens zwei Teilnehmern und einer Befehlseinheit, insbesondere Realtime Ethernet,
d a d u r c h g e k e n n z e i c h n e t,
dass bei der Befehlseinheit eine Schnittstelle verwendet wird, die aus wenigstens einem Eingangs-Register für alle Teilnehmer und wenigstens einem Ausgangs-Register für jeden der Teilnehmer eines Kommunikationssystems besteht, wobei Zugriffe der Teilnehmer auf die Befehlseinheit in dem Eingangs-Register empfangen und gespeichert werden und die Rückgabe der Befehlseinheit getrennt nach Teilnehmer in die jeweiligen Ausgangs-Register geschrieben und von dort an die Teilnehmer zurückgegeben werden.

2. System zum Durchführen eines Verfahrens nach Anspruch 1.

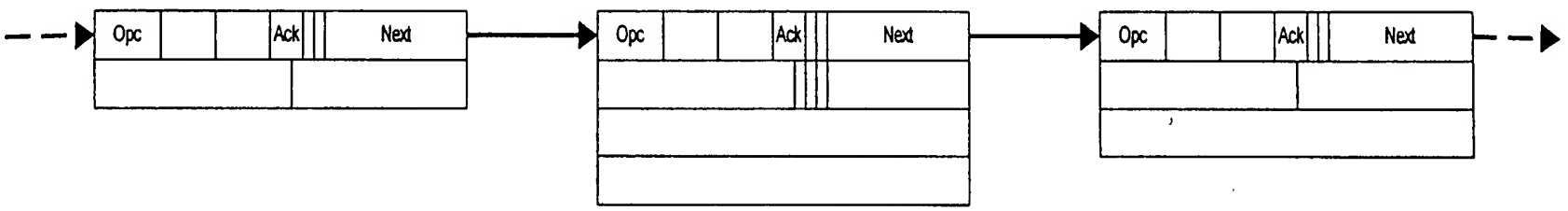

Bild 1: Beispiel einer Verkettung von Befehlsstrukturen

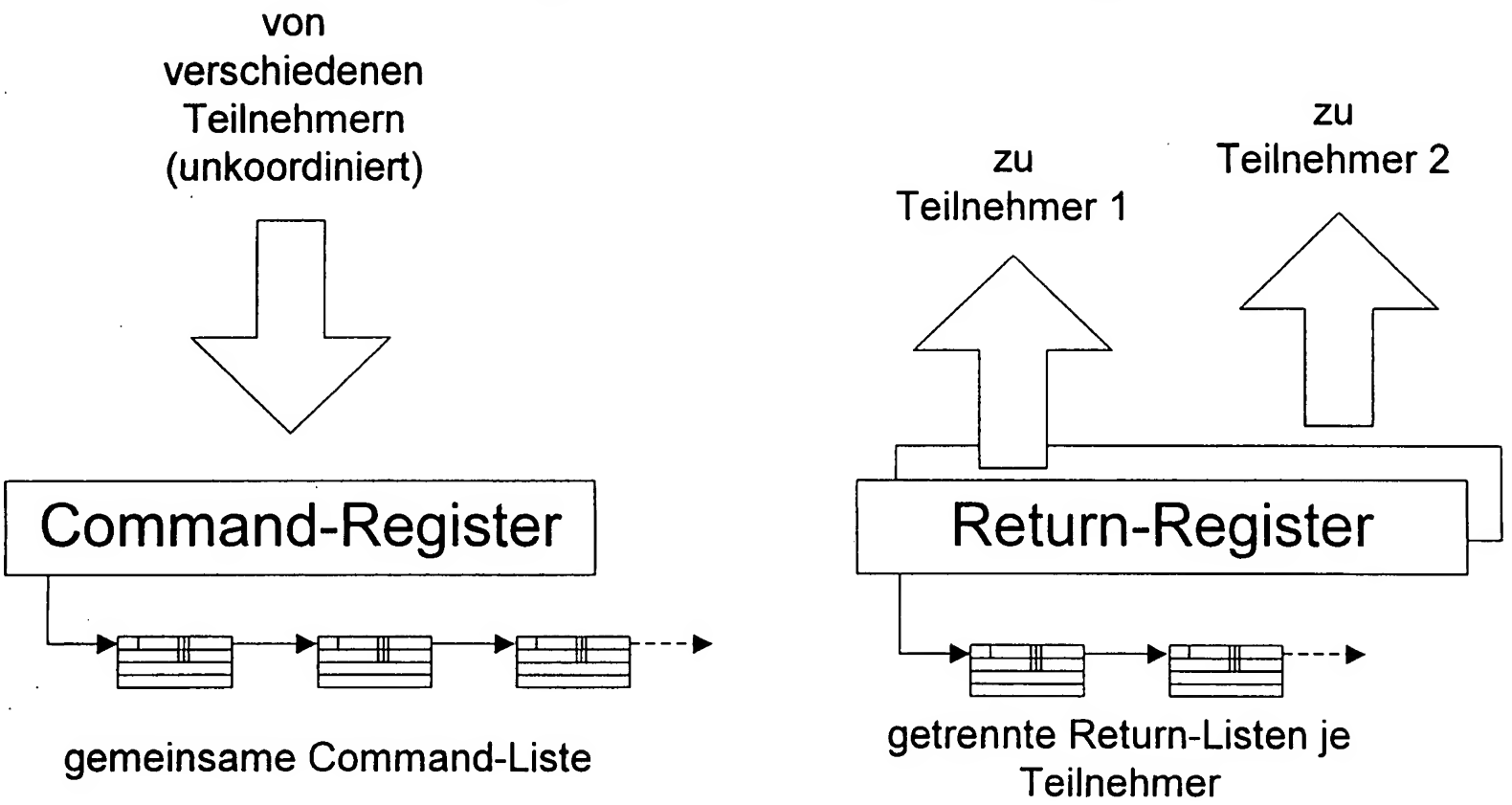

Bild 2: Aufbau des Command-Interface

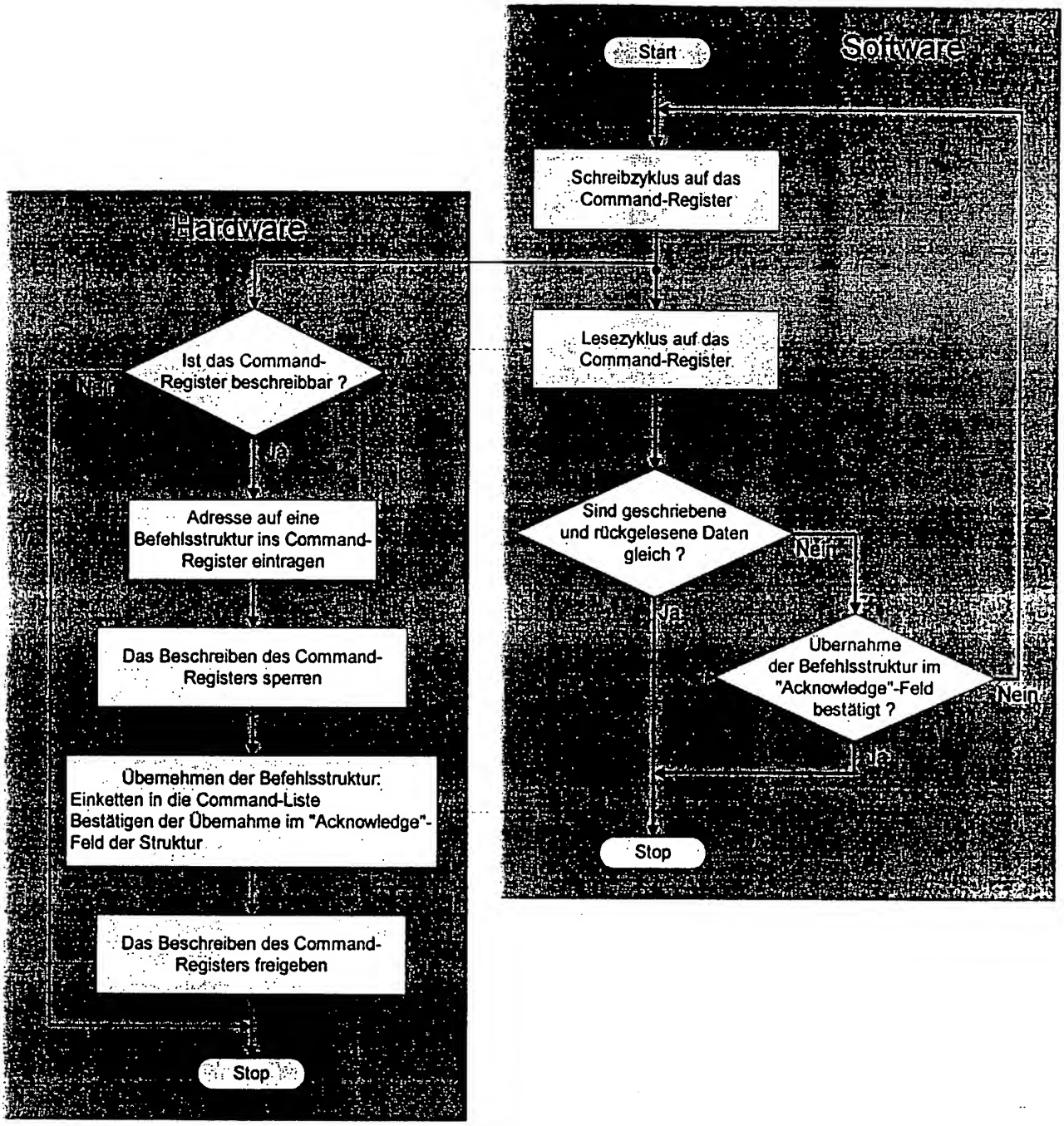

Bild 3: Hard- und Softwareablauf am Command-Interface